Advanced Functional Programming

Software Engineering Chair and Programming Systems Lab

Questions for *The Influence of Browsers on Evaluators or, Continuations to Program Web Services* by Christian Queinnec. It was presented at the ACM International Conference on Functional Programming 2000.

Web Development

1. Consider web programming without continuations. How does a naive implementation of the example from Section 2.2 look like? Sketch web pages, their embedded links, and the programs generating them.

2. Considering your experience from the previous question, why is programming with continunations better? Consider reusability of code in addition to general aspects (page-centric vs. program-centric). What is the function on the server-side where most of the magic is happening?

3. Given that HTTP is a stateless protocol, how does a naive web-shop application identify requests belonging to the same user? How do continutations help to solve this problem? What problems are shared by both approaches?

4. The server forks a thread for every request. Why would a developer in addition use `fork` in its server-side program?

Semantics of `callcc` and `fork`

1. The following reduction rules specify the operational semantics of `callcc`:

$$\begin{aligned} E[\texttt{callcc}\ F] &\rightarrow E[F(\lambda x.\texttt{abort}\, E[x])] \\ E[\texttt{abort}\ M] &\rightarrow M \end{aligned}$$

   Try to understand these rules. What is the role of the auxiliary construct `abort`? What are the types of `callcc` and `abort`?

2. Consider the following naive encoding of exceptions via `callcc` and vice versa:

$$\begin{aligned} \texttt{try}\ M\ \texttt{handle}\ x \rightarrow N &\implies \texttt{callcc}\ (\lambda k.M[\texttt{raise} := \lambda x.k(N)]) \\ \texttt{callcc}\ F &\implies \texttt{try}\ F(\lambda x.\texttt{raise}\ x)\ \texttt{handle}\ x \rightarrow x \end{aligned}$$

   Why does neither work in general? Find examples where they break down.

3. The `fork` operation used in the paper has the following semantics:

$$E[\texttt{fork}\ M\ N] \quad \rightarrow \quad E[M] \mid E[N]$$

It is more common to have thread creation of the following form:

$$E[\texttt{spawn}\ M] \quad\rightarrow\quad E[()]\,|\,M$$

What is the difference? How can both be expressed with each other (Hint: you need `callcc` and `abort`)?

Homework

Next Tuesday we will present papers and ideas for talks. Please think about the papers and topics that you are most interested in. You are also welcome to suggest a topic for your talk that we haven't read about.

We also want to find a date for all talks, so please bring your calendar. We are planning to reserve two consecutive days for talks, probably after the exams and after the end of lectures.